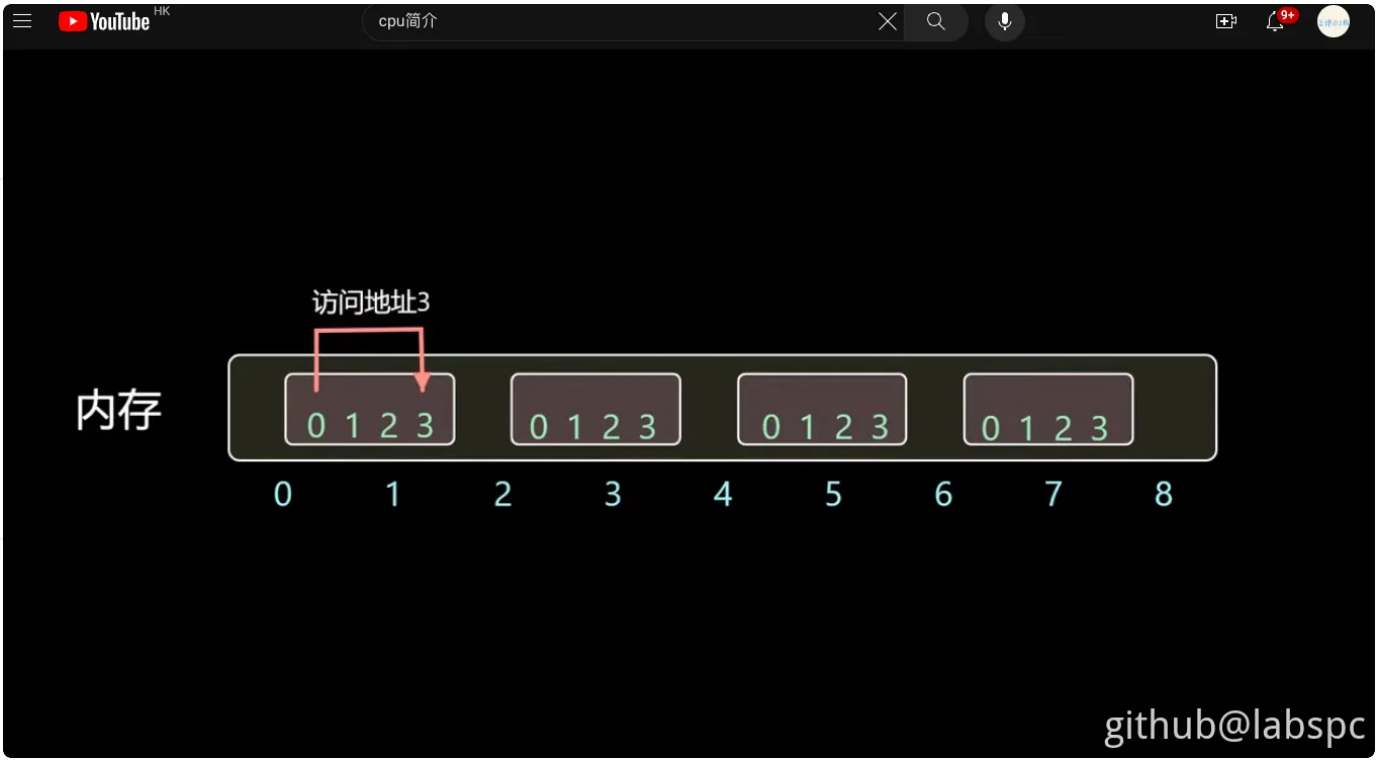
0.9 进程与线程

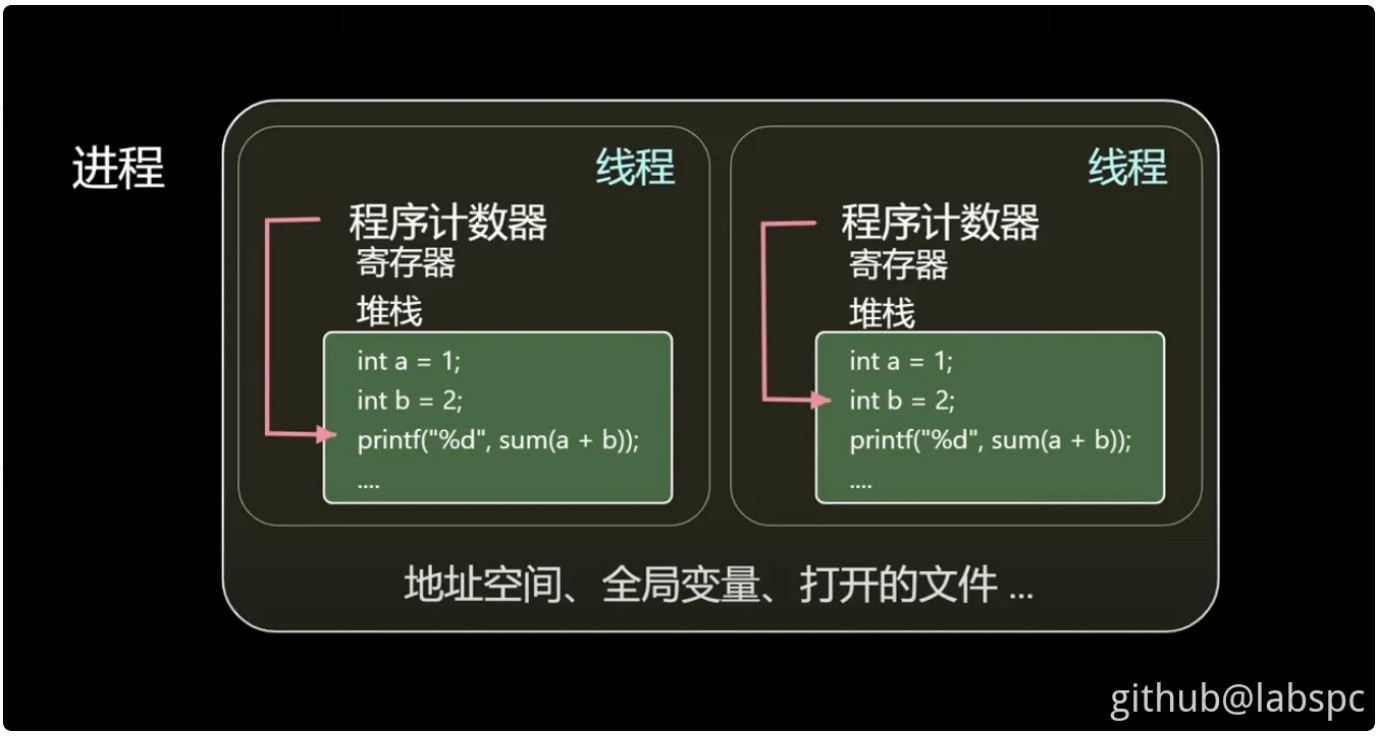
https://youtu.be/e3JQOgKw9BA?si=YudyAlTHoZ4kxHbz

cpu从硬盘中读取程序到内存，那这个在内存中可执行程序实例就叫做进程。

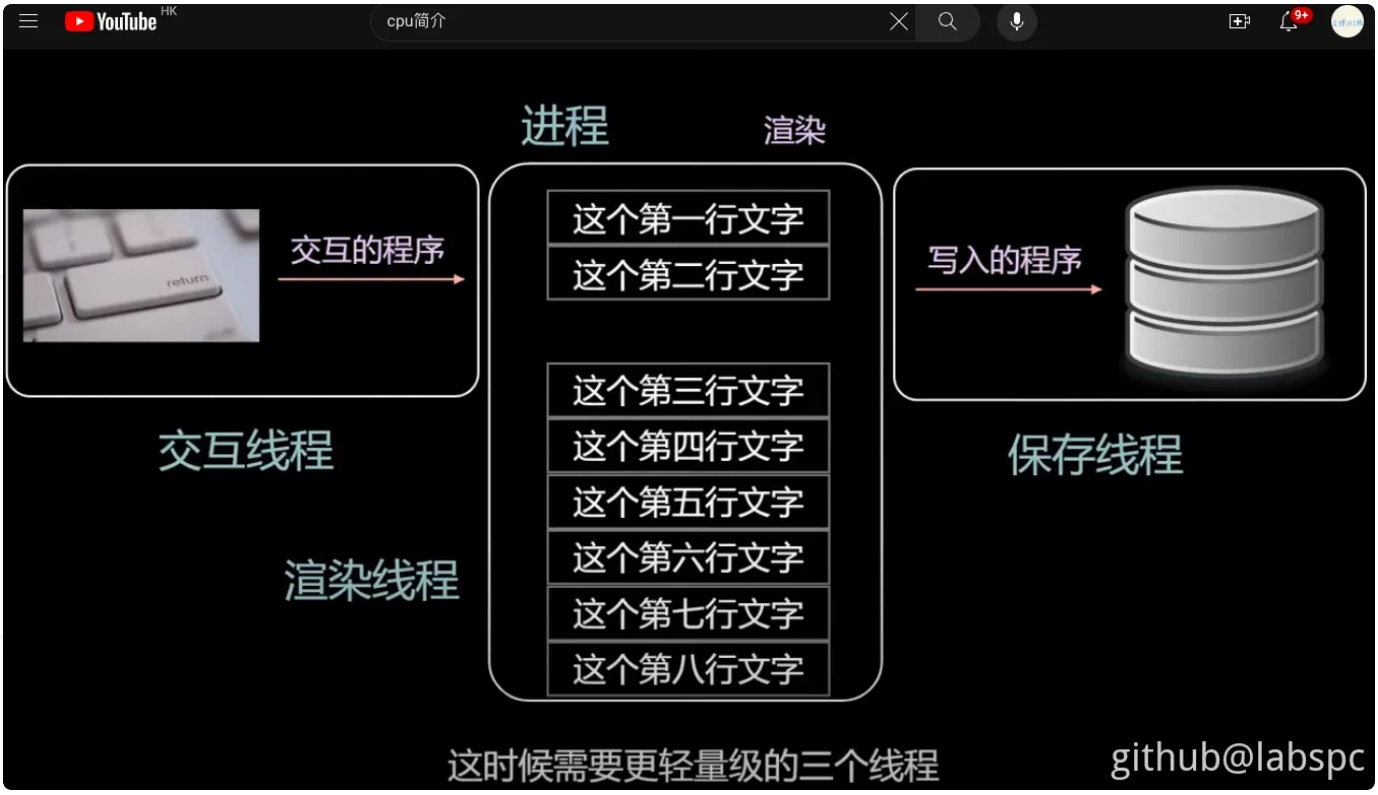
内存中任何一个地方都有相应的地址方便访问，而在内存中的每个进程，自己内部都有一个虚拟独立的地址空间，在进程内，就可以根据虚拟地址来访问。**进程是程序执行的完整单位，而且大部分时间都是在进程内，那进程间就需要进程间通信IPC（Inter Process Communication）。 每个进程都以为自己独占整个内存，不需要关心其他进程的实际位置，这样就很好把进程分割开了。**



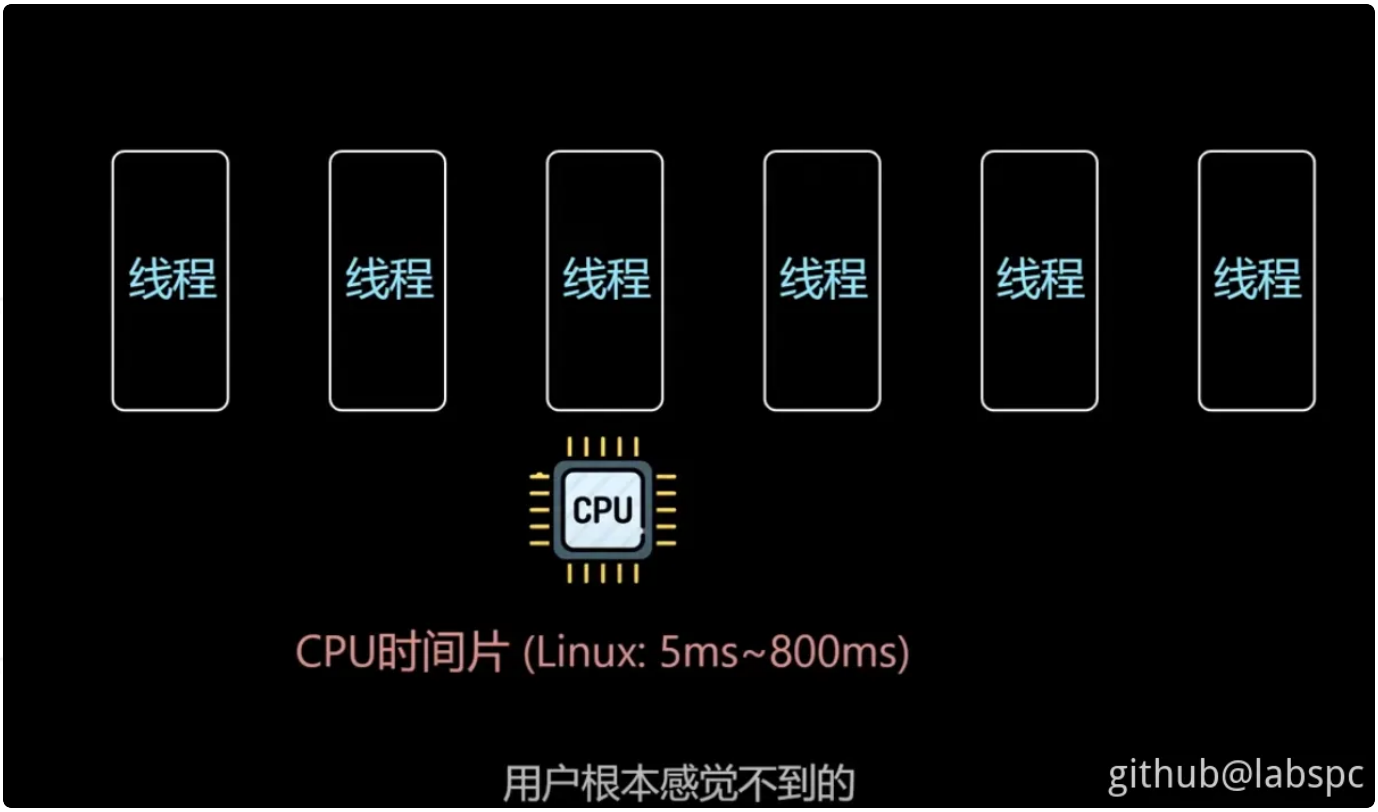
每个进程首先有加载的程序，通常只有一个程序计数器，记录当前程序执行的位置，会按照程序顺序计算，这里的一个执行流就是一个线程。如果有多个线程的话，就需要多个程序计数器，每个线程会独自运行。除此之外，每个线程还有寄存器、堆栈等程序运行时的状态信息。线程间共享的则有地址空间、全局变量、打开的文件等等信息，



为什么在有进程后，还要有更小的线程？ 线程是并行的最小单位。

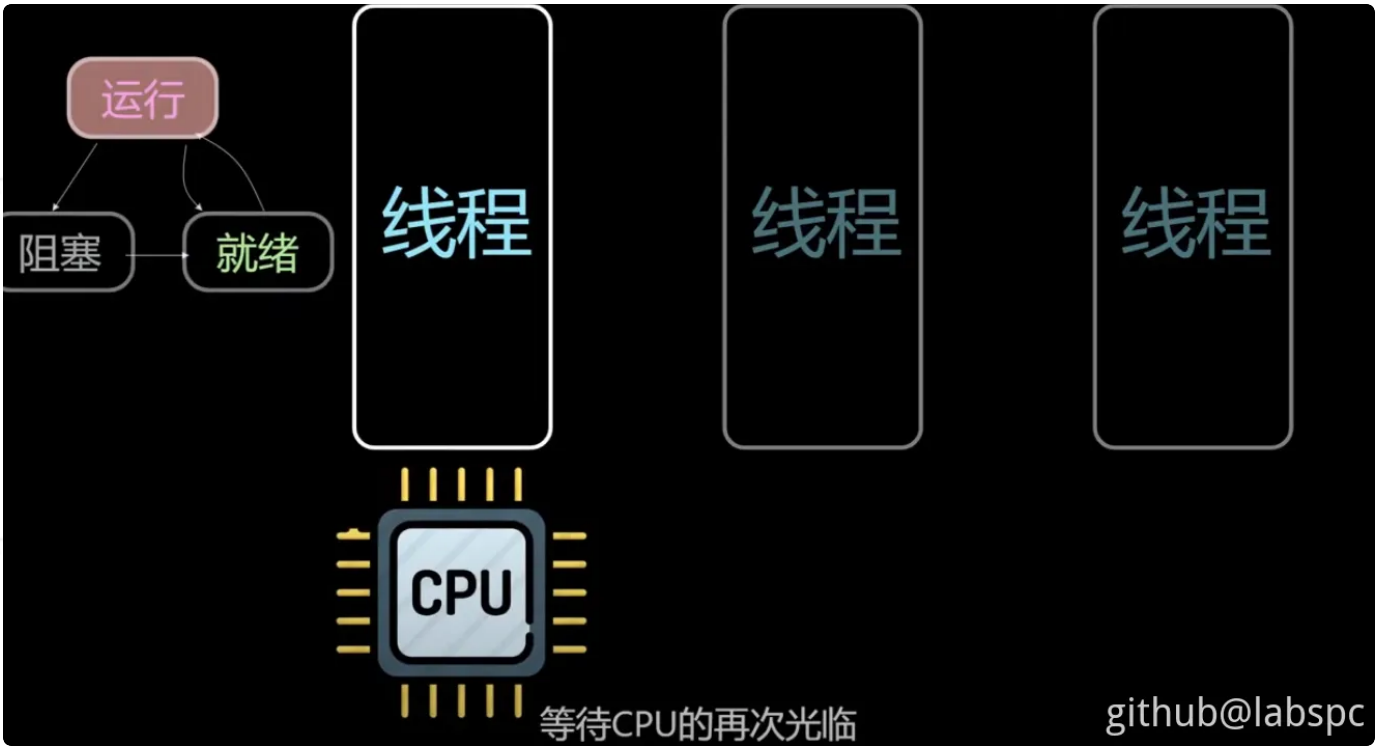


CPU如果只有一个单核，那就需要对每个线程进行轮流执行，每个单个计算的时间成为一个CPU时间片。



对线程来说，存在等待CPU的时候，称为就绪状态， 一旦过来执行，就称为运行状态，走了之后又称为就绪状态。假如线程在执行中，程序向硬盘发送访问请求，然后等待，这时CPU就变成空转，所以线程就变成阻塞状态。 CPU转而执行其他线程，等到硬盘的数据回复，线程从阻塞状态回到就绪状态，这时再等待CPU的再次执行。

注意：在进程开始阻塞时，CPU并不会等待，转而去执行别的线程。


运行
阻塞
就绪
线程
线程
线程
CPU
等待CPU的再次光临
github@labspc